NYU TANDON SCHOOL OF ENGINEERING

Network Security

Review of Selected Materials

Phillip Mak
pmak@nyu.edu

Logistics

- Midterm -- Saturday, 15 March, starting time between 1 – 3 PM ET
  - 2 hours long
  - Covers all topics except for Lesson 4 slides 41+
  - The midterm exam is open book, open notes, open VM, and open Internet. However, it must be performed individually -- you may not collaborate or discuss the exam with anyone until the exam grade is released. Please see the definitions of cheating and unauthorized collaboration in the Student Code of Conduct.
  - Example of unauthorized collaboration is posting questions or reading answers with other students during the exam, or though a forum or question and answer site.
  - Usage of ChatGPT or any other AI content generation tools would be considered Plagiarism. Plagiarism checkers such as Turnitin and similar will be used
  - The timer does not stop even if you submit or close the window
- No bonus exercises this week.

- Excused Absence. If you get sick, don't take the exam. Get documentation.
  - https://engineering.nyu.edu/student-life/office-student-affairs/policies

- Here are the things you should be working on:
  - Review the sample midterm problems
  - Review the Crypto lecture
  - HW1 & , Labs 1 & 2

Objectives

- Review selected materials for the midterm exam
  - Lab 1 / Lab 2 / HW #2
  - Lesson 1-3 and 5
  - (No lesson 4 – see last week's video)

Lab 1 Review

Connection flooding: Overwhelming connection queue w/ SYN flood

Recall client sends SYN packet with initial seq. number when initiating a connection.

TCP on server machine allocates memory on its connection queue, to track the status of the new half-open connection.

For each half-open connection, server waits for ACK segment, using a timeout that is often > 1 minute

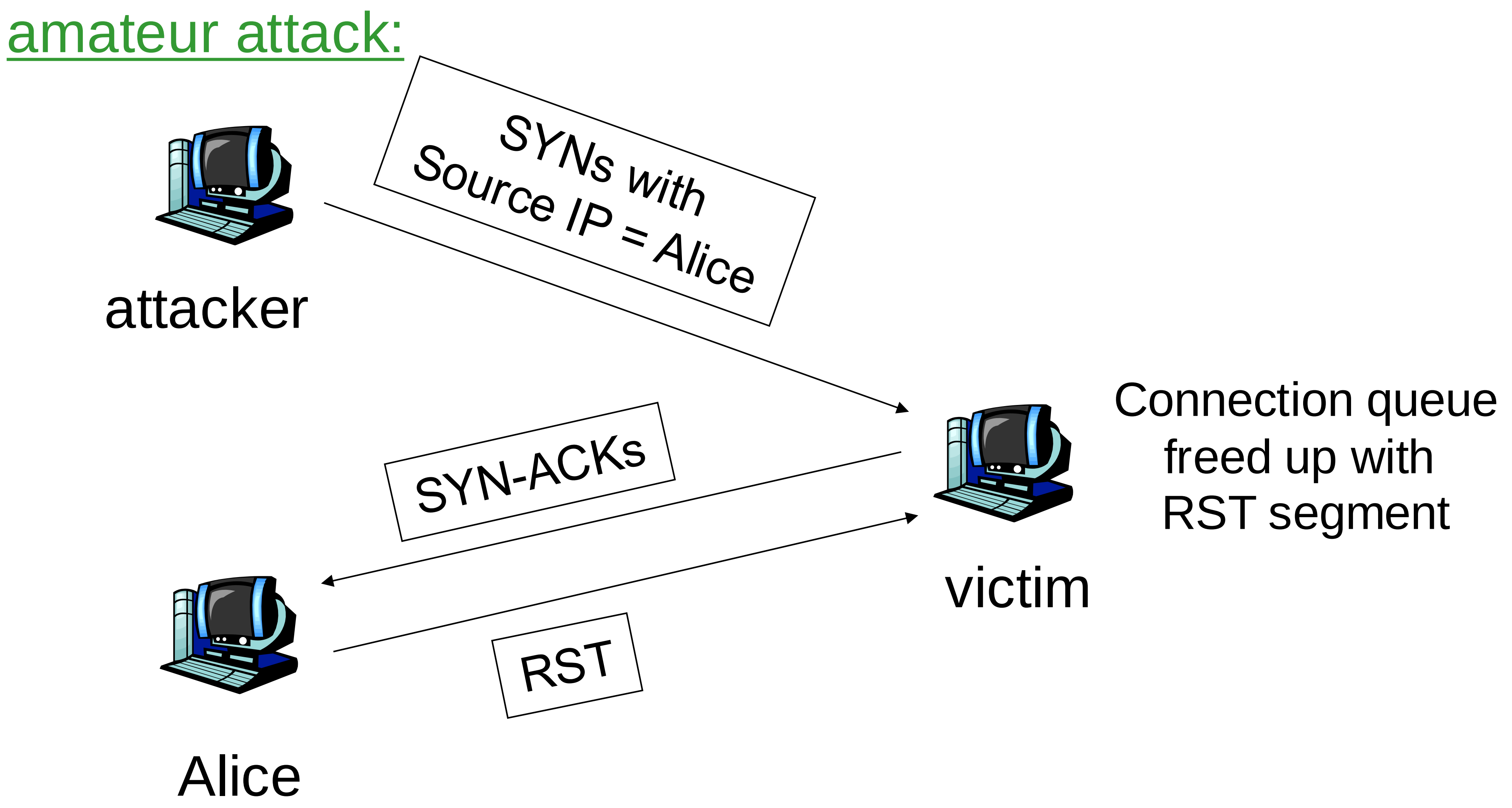
Attack: Send many SYN packets, filling connection queue with half-open connections.

Can spoof source IP address!

When connection queue is exhausted, no new connections can be initiated by legit users.

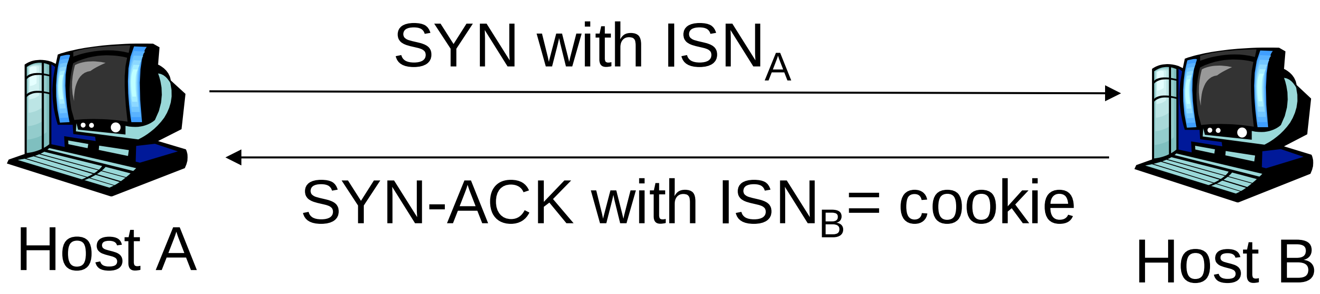
Need to know of open port on victim's machine: Port scanning.

DoS: Overwhelming connection queue with SYN flood



Expert attack: Use multiple source IP addresses, each from unresponsive addresses.

SYN flood defense: SYN cookies (1)



- When SYN segment arrives, host B calculates function (hash) based on:
  - Apache example: Source and destination IP addresses and port numbers, and a secret number
- Host B uses resulting "cookie" for its initial seq # (ISN) in SYNACK
- Host B does not allocate anything to half-open connection:
  - Does not remember A's ISN
  - Does not remember cookie

SYN flood defense: SYN cookies (2)

If SYN is legitimate
Host A returns ACK

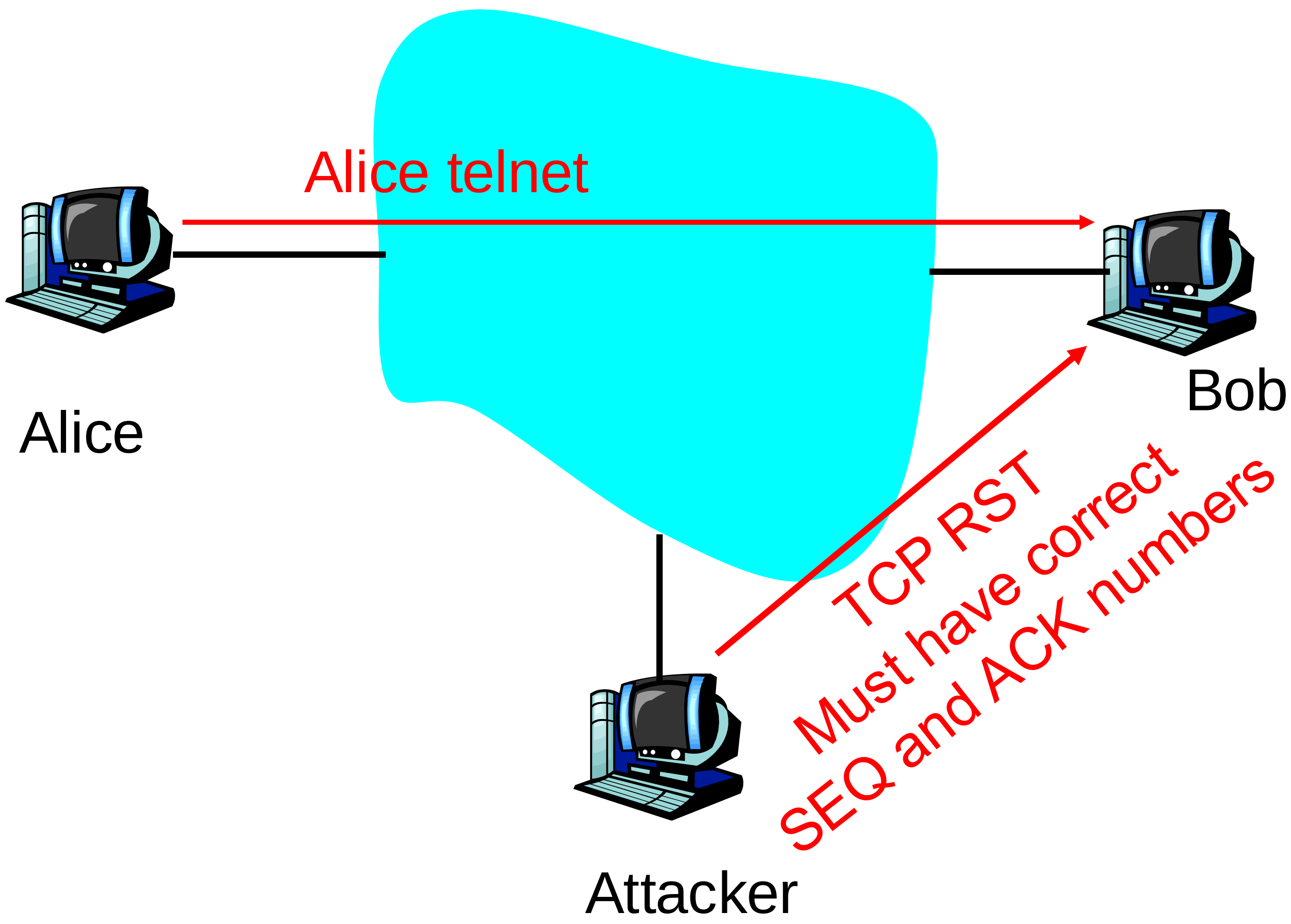
Host B computes same function, verifies function = ACK # in ACK segment
Host B creates socket for connection

Legit connection established without the need for half-open connections

If SYN-flood attack with spoofed IP address
No ACK comes back to B for connection.

No problem: B is not waiting for an ACK

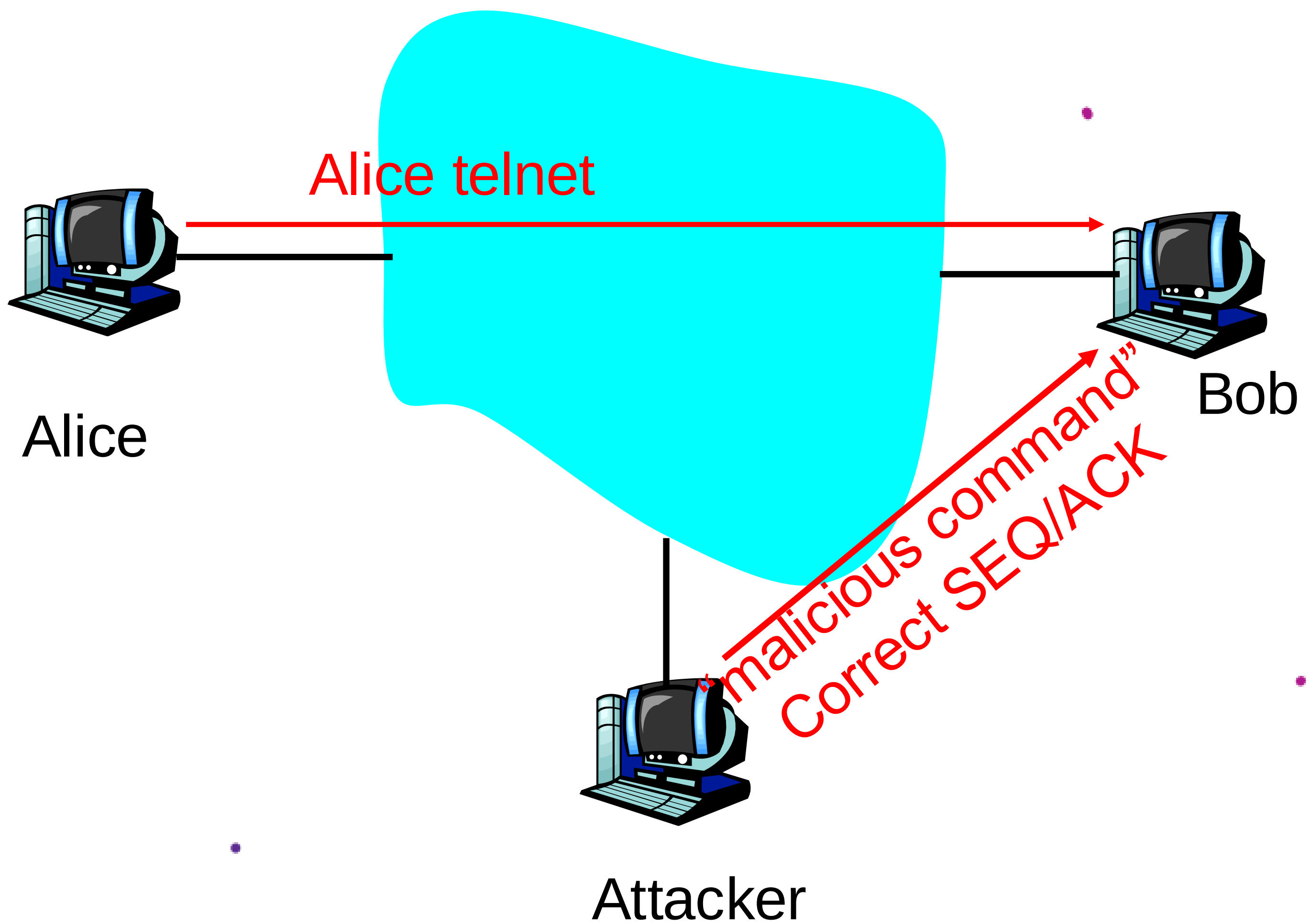
TCP RST Attack



- Attacker can break the TCP connection by sending a TCP RST
- Must match the SEQ and ACK Numbers

Session hijacking

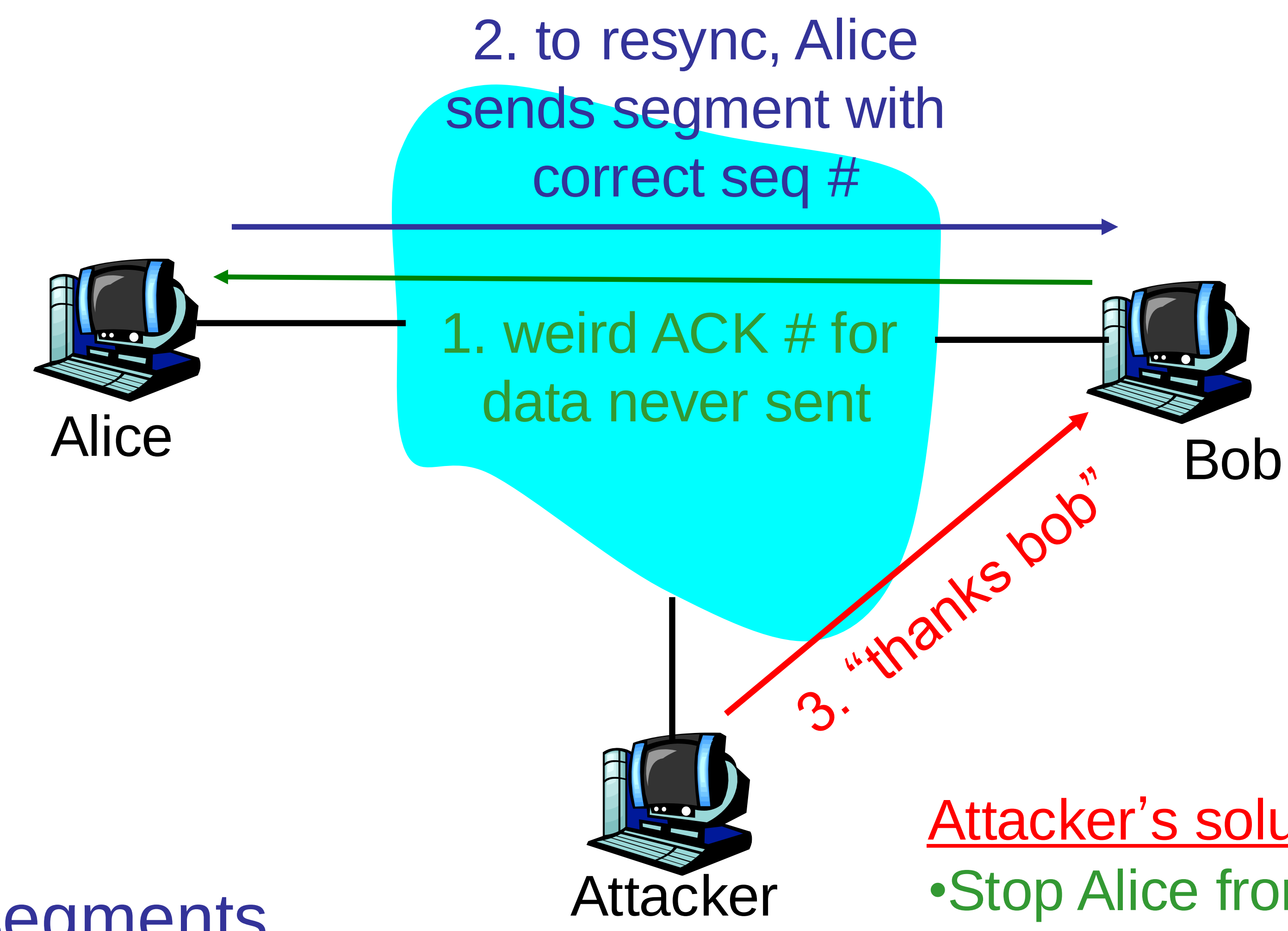
- Take control of one side of a TCP connection
- Marriage of sniffing and spoofing



Session hijacking: The details

- Attacker is on segment where traffic passes from Alice to Bob
  - Attacker sniffs packets
  - Sees TCP packets between Bob and Alice and their sequence numbers
- Attacker jumps in, sending TCP packets to Bob; source IP address = Alice's IP address
  - Bob now obeys commands sent by attacker, thinking they were sent by Alice
- Principal defense: encryption w/ auth protocol
  - Attacker does not have keys to encrypt and insert meaningful traffic

Session hijacking: limitation



Bob is getting segments from attacker and Alice. Source IP address same, but seq #’s different. Bob likely drops connection.

Attacker’s solution:

- Stop Alice from communicating with Bob
- Poison the ARP Cache
  - Send unsolicited ARP replies to Alice and Bob with non-existent MAC addresses
  - Overwrite IP-to-MAC ARP tables so Alice’s segments will not reach Bob and vice-versa
  - But attacker continues to hear Bob’s segments, communicates with Bob

Lab 2 Review

Q1: Use `sniff()` to capture packets. Learn how to use `filter=`

Q2: Spoof ICMP echo request packets

Q3: Write an ICMP traceroute program

Q4: sniff() icmp echo-request, and spoof echo-replies

- Q4.2: ping 1.2.3.4
- Q4.3: ping 10.9.0.99 (does not work, explain why)
- Q4.4: ping 8.8.8.8

Q5: extra credit. Make Q4.3 (ping 10.9.0.99) work by using ARP cache poisoning (write a scapy program to perform ARP cache poisioning)

Risk Matrix

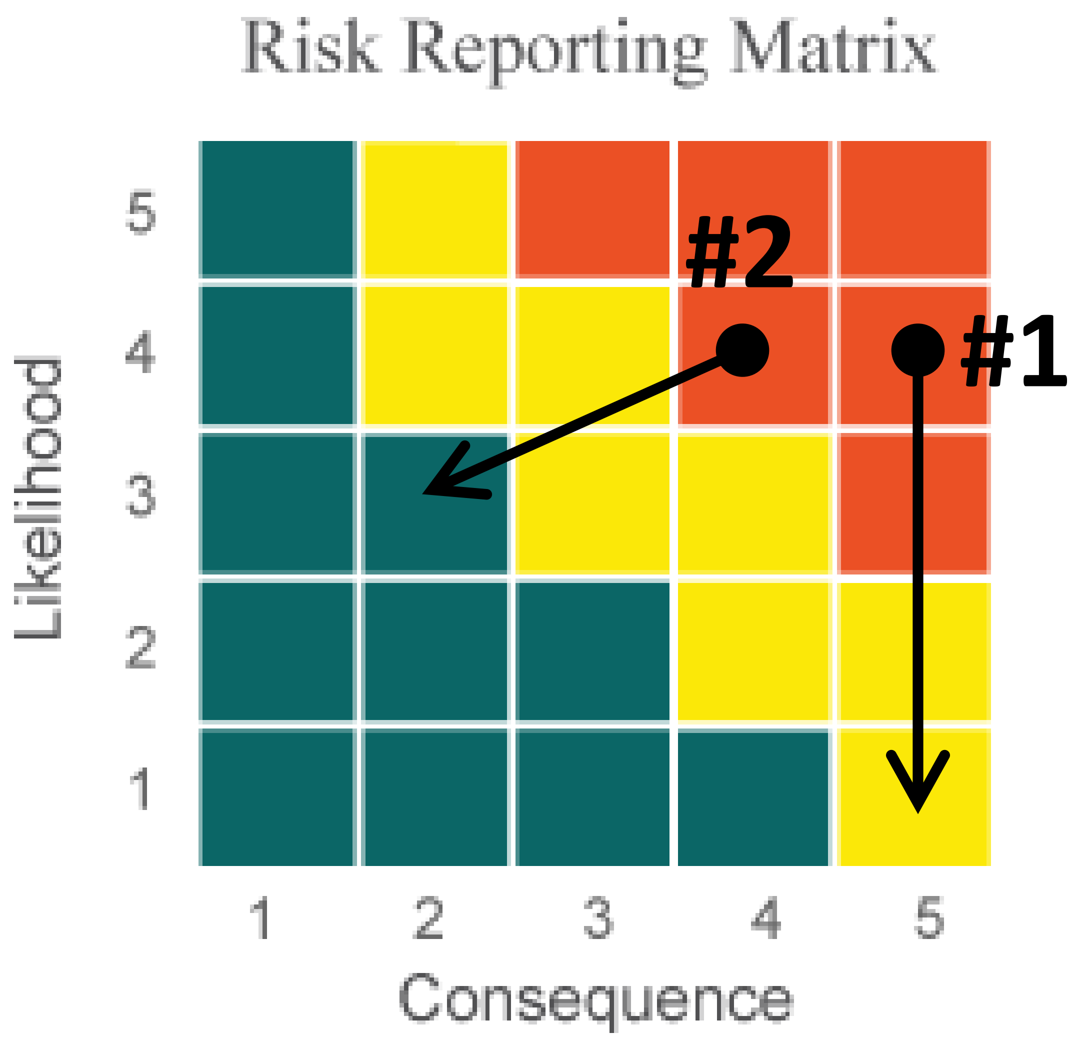
Level	Likelihood	Probability of Occurrence
5	Near Certainty	~ 90%
4	Highly Likely	~ 70%
3	Likely	~ 50%
2	Low Likelihood	~ 30%
1	Not Likely	~ 10%

Level	Consequences
5	Severe
4	Significant
3	Moderate
2	Minor
1	Minimal or no consequences

Exercise A

- What is residual risk?

Mitigating Risk



Example Risk #1: The software is really buggy and will likely have buffer overflow vulnerabilities. *Reduce the likelihood of this risk by spending more resources to reduce defects.*

Example Risk #2: There's a 70% chance the website will be hacked and 1 million credit card numbers will be lost. *Reduce consequences by not storing full credit card numbers. Likelihood reduced by adding a web firewall.*

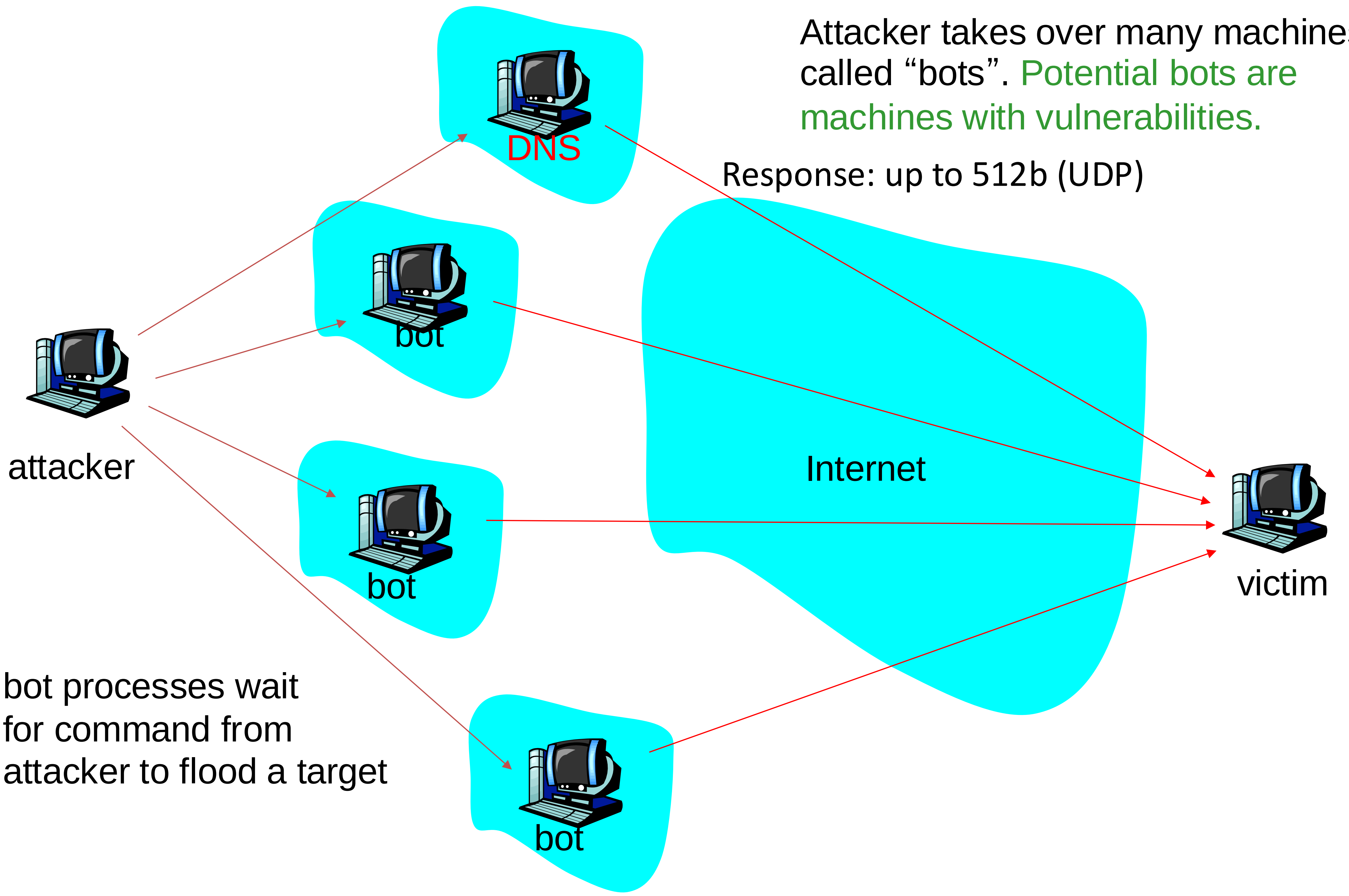
Residual risk is the remaining risk after mitigations

Quantitative Risk Assessment Example

- Fire Damage to a building:
  - –Asset Value: value of the building - $750,000
  - –Single Loss Expectancy (SLE: Asset Value x Exposure Factor) - $250,000 (damage caused by the fire)
  - –Annualized Rate of Occurrence (ARO) - .05 (5% chance every year that there will be a fire)
  - –Annualized Loss Expectancy (ALE: $250,000 x .05) = $12,500

- So does a fire alarm system which costs $5000/year to maintain and $15k to install initially worth it?

Distributed DoS: DDos

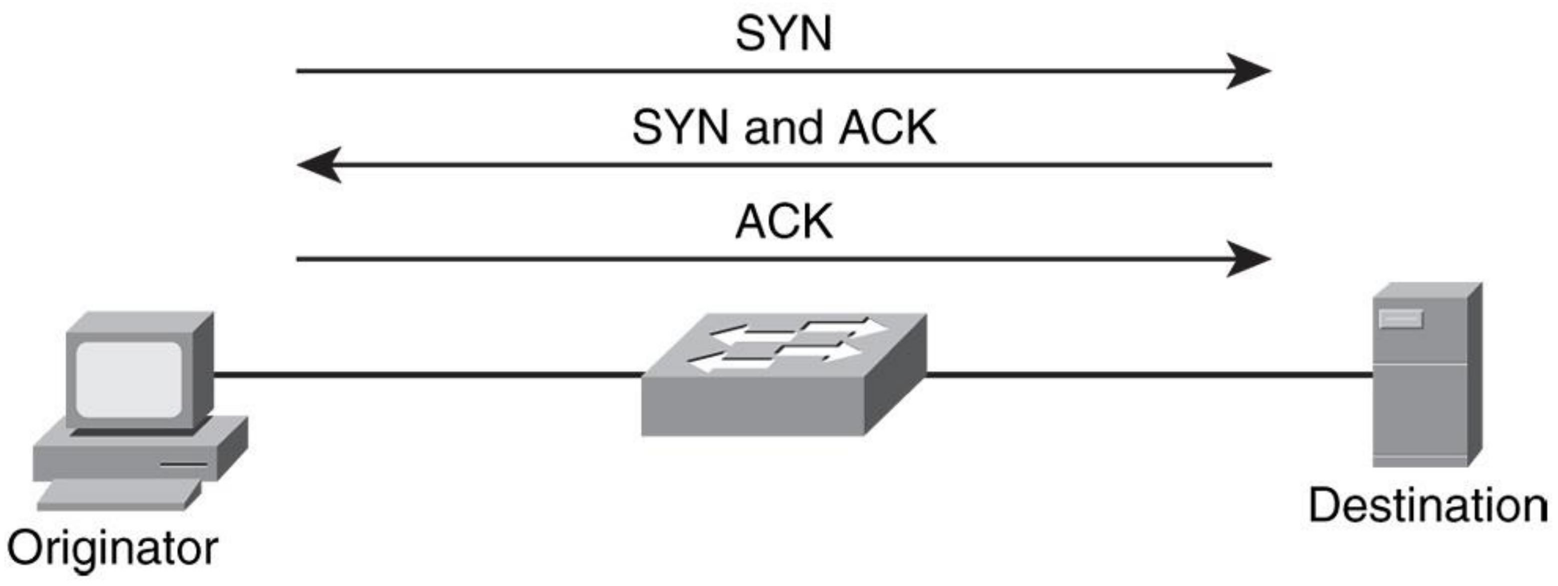


Attacker takes over many machines, called “bots”. Potential bots are machines with vulnerabilities.

bot processes wait for command from attacker to flood a target

Port Scanning

- Port scanners send TCP and UDP packets to various ports to determine if a process is active
  - TCP 80 (web server)
  - TCP 23 (telnet server)
  - UDP 53 (DNS server)
- TCP scanning based on 3 way handshake

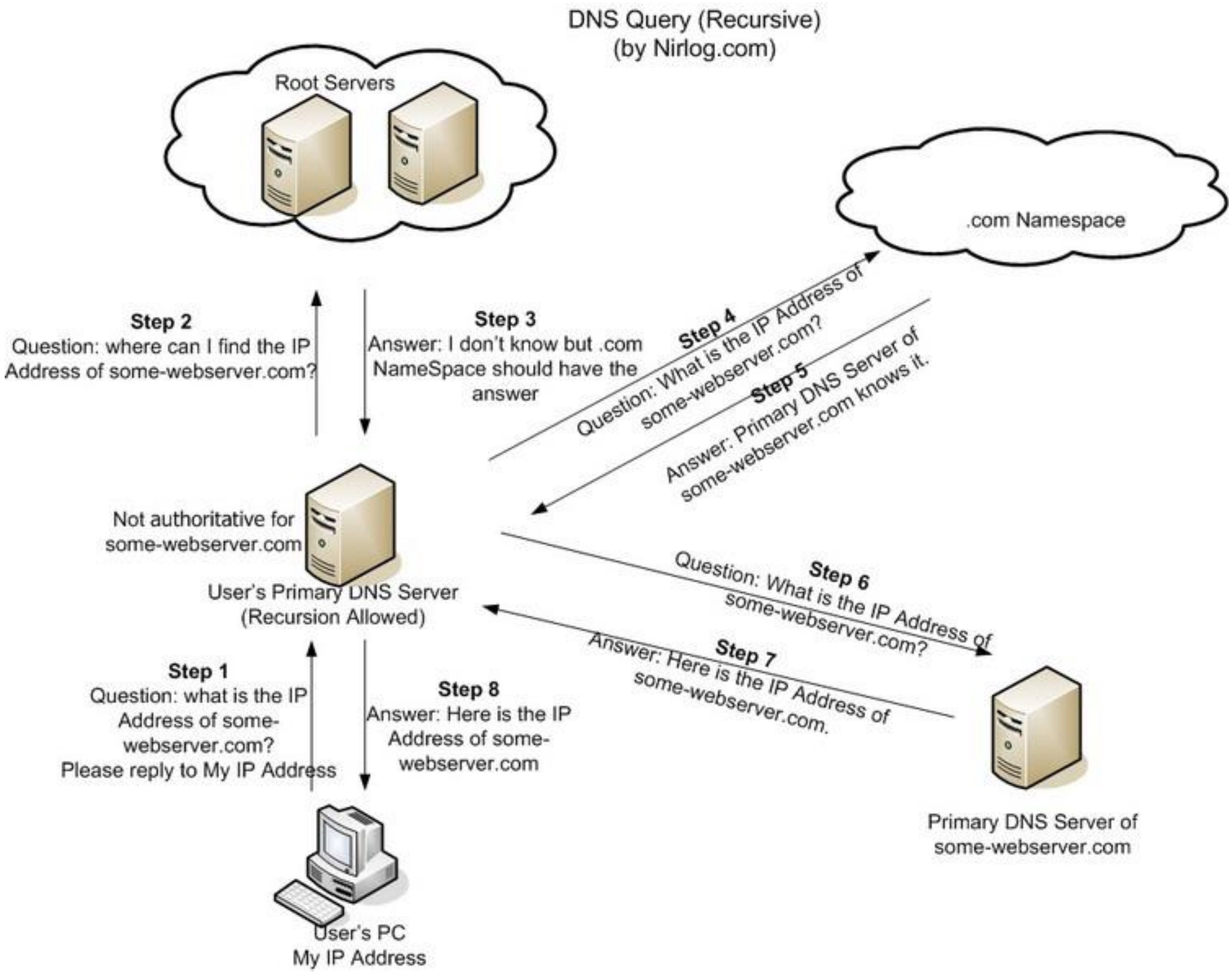


Exercise B

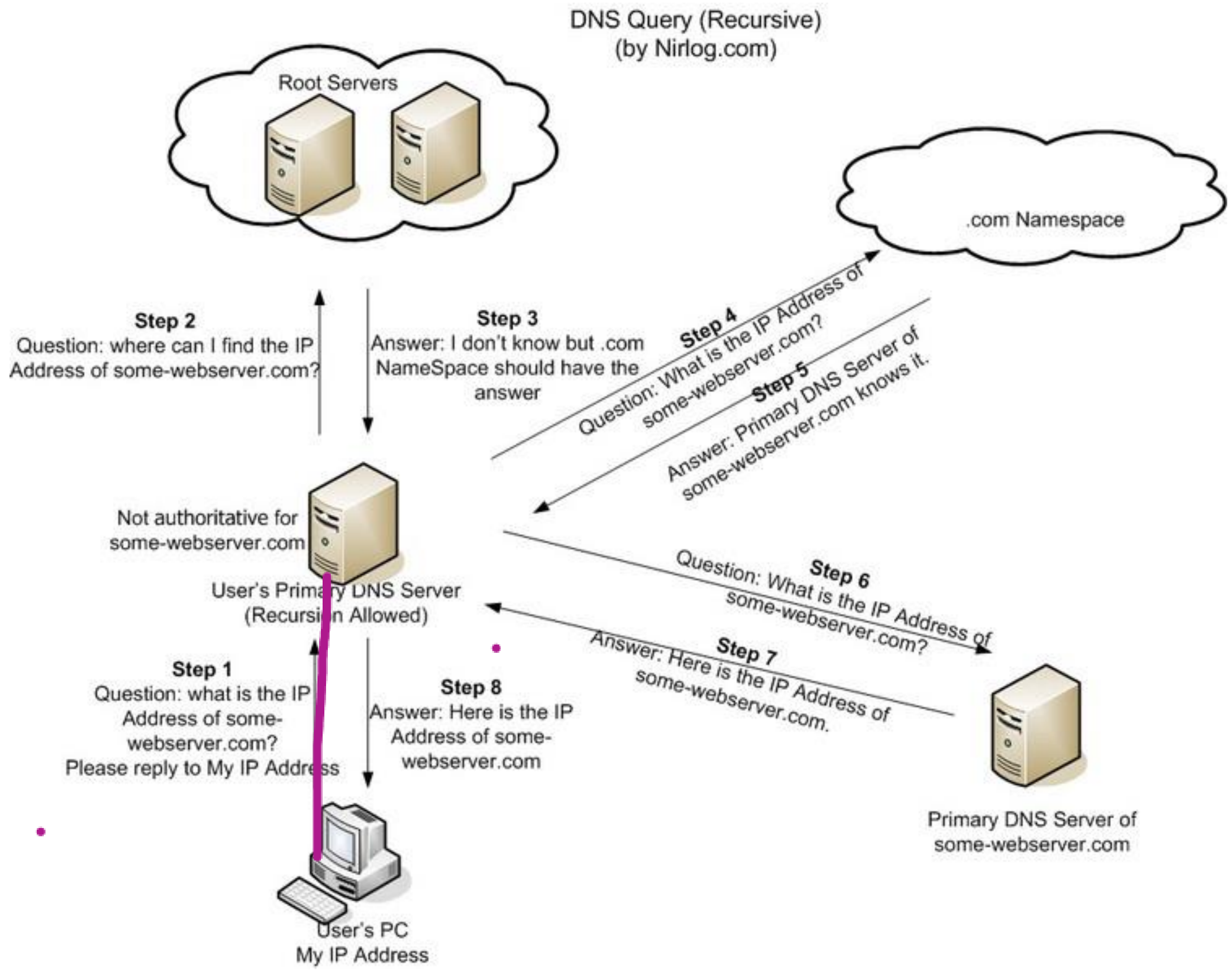
- What are the possible responses to a TCP SYN packet, and the reasons why for each?

- UDP?

Interlude: How DNS Works



Exercise C



Suppose an attacker wants to perform DNS cache poisoning so that the website www.nytimes.com to be diverted to www.evil.com

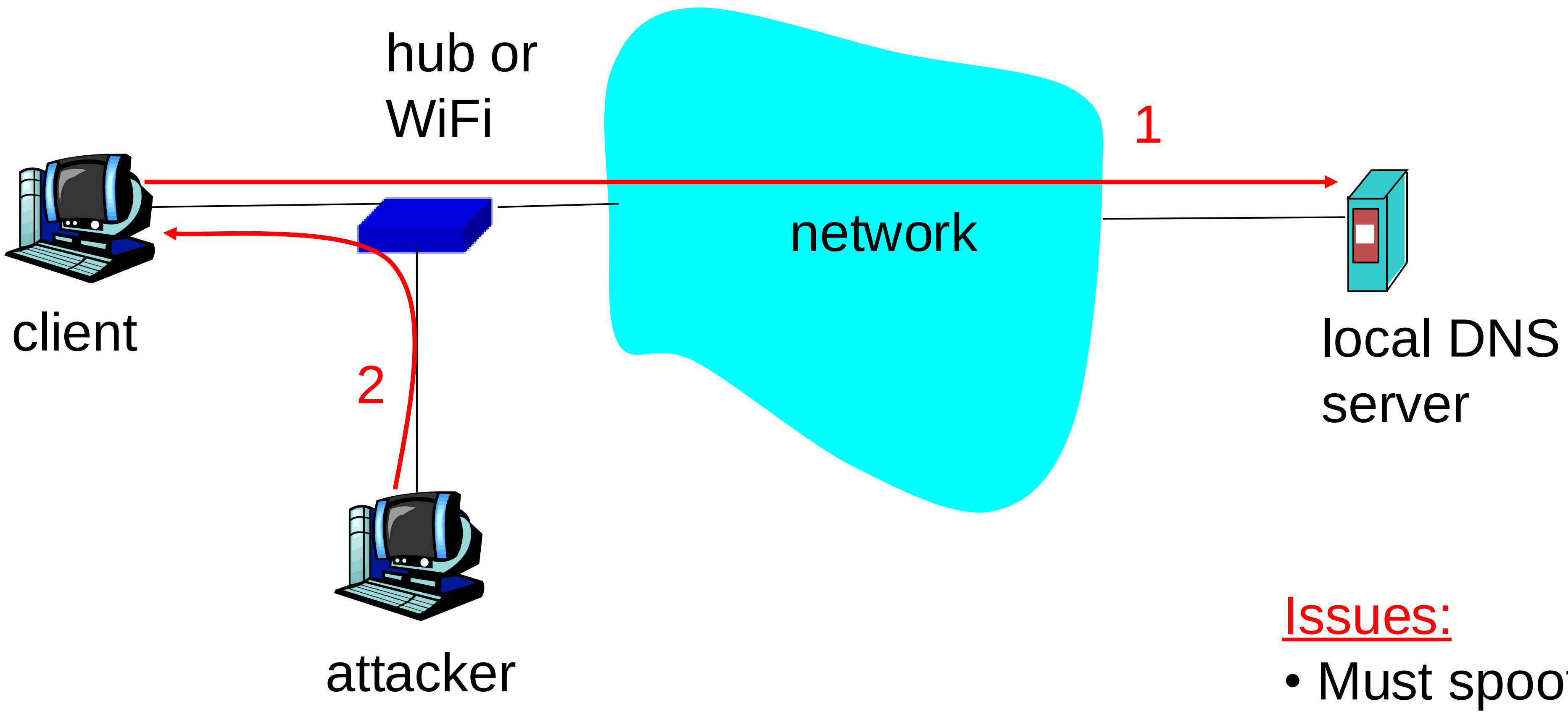
a. [2 pts] If DNS cache poisoning was successful, would the user's browser show www.nytimes.com or www.evil.com? Explain.

b. [4 pts] Suppose an attacker is deciding between attempting to spoof the DNS response on Step 7, or to spoof the DNS response on Step 8. Explain the difficulty of performing **each** of these attacks.

c. [4 pts] Explain which users will be affected if the attacker successfully spoofs Step 7 as compared to if the attacker successfully spoofs Step 8.

d. Which step would the attacker spoof to affect ALL users of nytimes.com for Verizion FIOS including science.nytimes.com and policitcs.nytimes.com

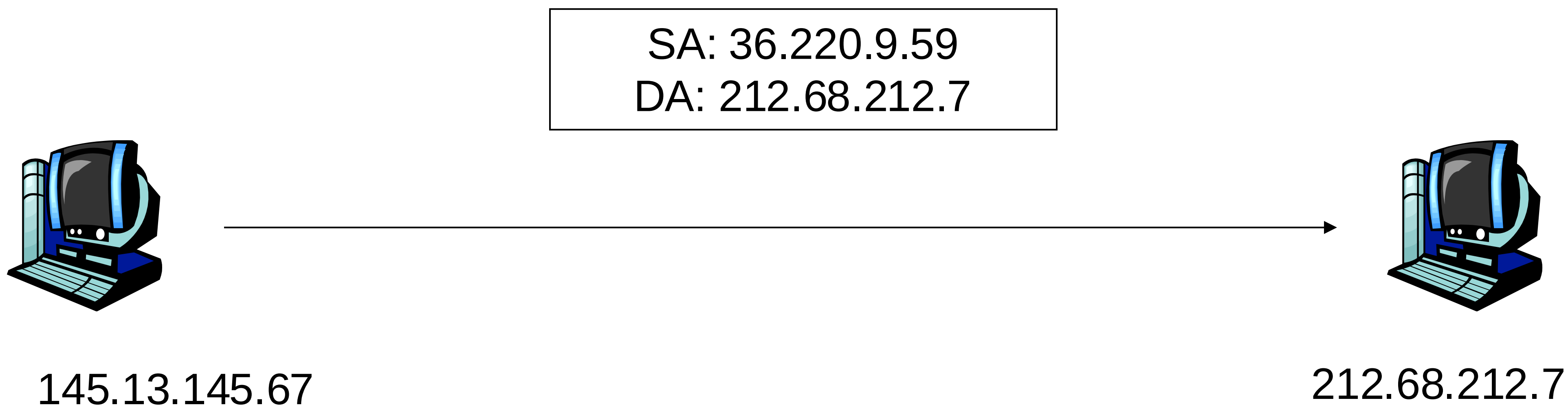
DNS attack: redirecting



1. Client sends DNS query to its local DNS server; sniffed by attacker
2. Attacker responds with bogus DNS reply

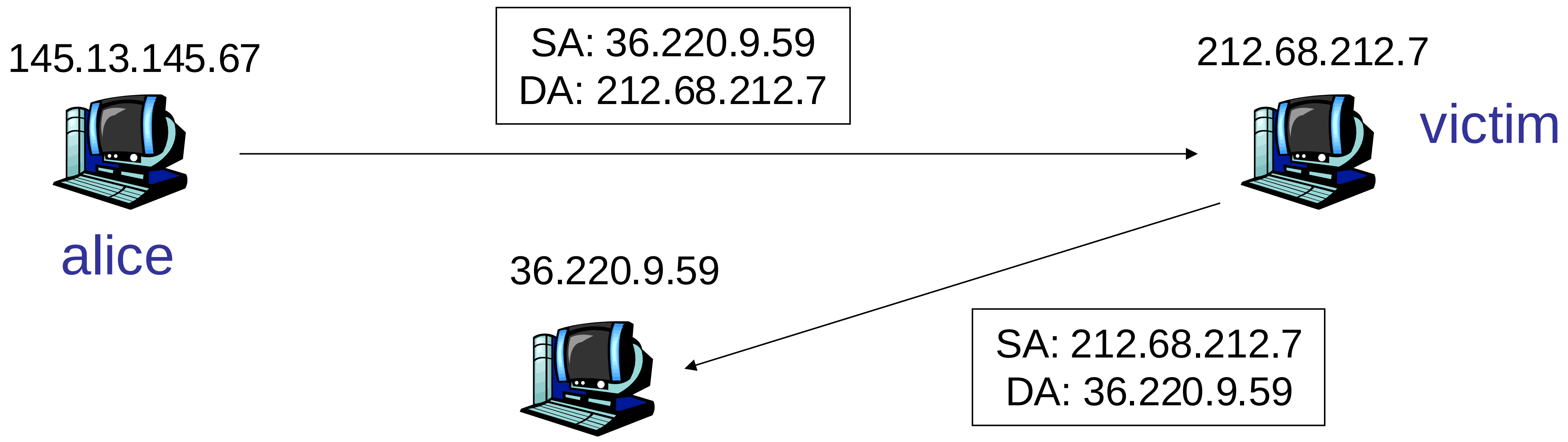
Issues:
- Must spoof IP address: set to local DNS server *(easy)*
- Must match reply ID with request ID *(easy if on the same LAN) – transaction ID*
- May need to stop reply from the local DNS server *(harder)*

IP address spoofing (1)



- Attacker doesn’t want actions traced back
- Simply re-configure IP address in Windows or Unix.
- Or enter spoofed address in an application
  - e.g., decoy packets with Nmap

IP address spoofing (2)



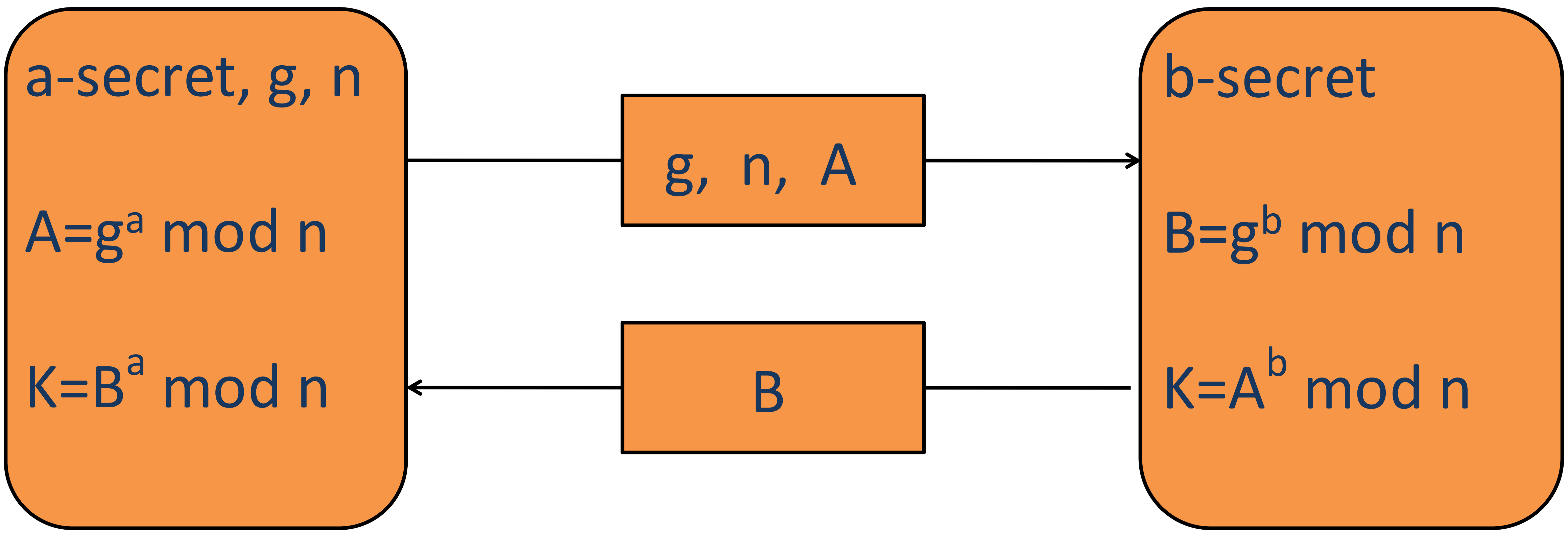
- But attacker cannot interact with victim.
  - Unless attacker is on path between victim and spoofed address.

IP spoofing with TCP?

- Can an attacker make a TCP connection to server with a spoofed IP address?
- Not easy: SYN-ACK and any subsequent packets sent to spoofed address.
- If attacker can guess initial sequence number, can attempt to send commands
  - Send ACK with spoofed IP and correct seq #, say, one second after SYN
- But TCP uses random initial sequence numbers.

Diffie-Hellman

- Allows two entities to agree on shared key.
  - But does not provide encryption
- n is a large prime; g is a number less than n.
  - n and g are made public



Trudy – sees g, n, A, B, but cannot decipher K

Diffie-Hellman (cont)

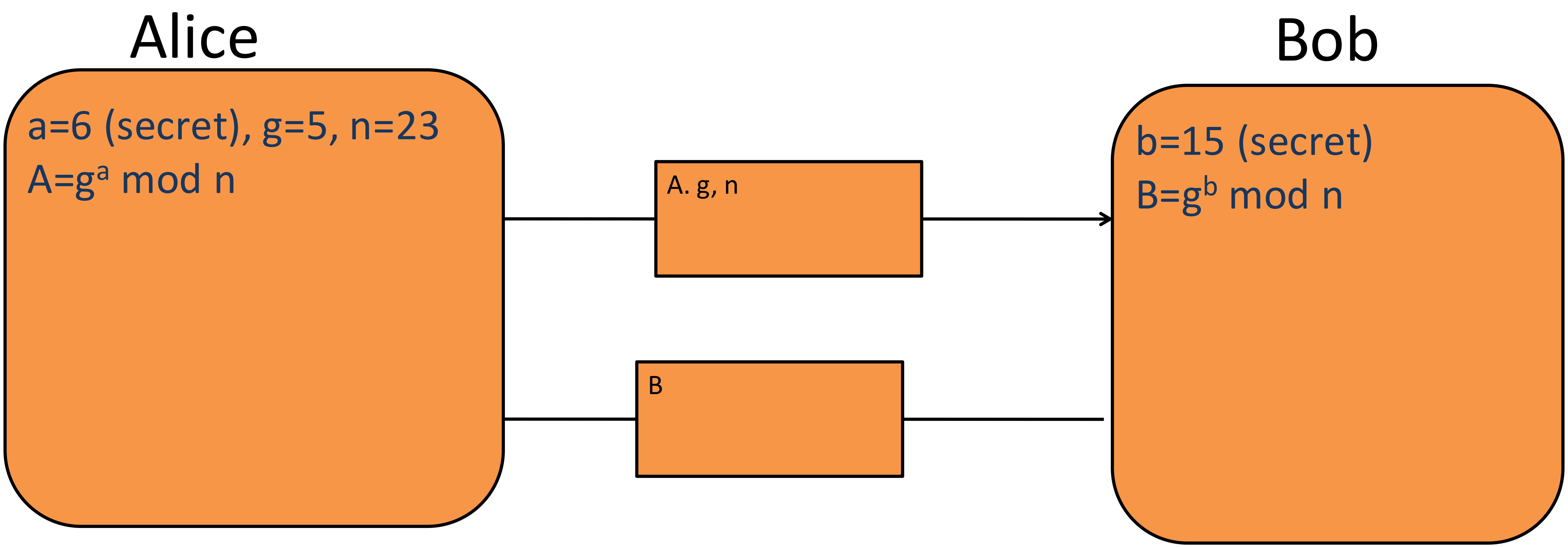
- Alice and Bob agree to use a prime number n=23 and base g=5.
- Alice chooses a secret integer a=6, then sends Bob $A = g^a \bmod n$
  - $A = 5^6 \bmod 23$
- Bob chooses a secret integer b=15, then sends Alice $B = g^b \bmod n$
  - $B = 5^{15} \bmod 23$
- Alice computes $s = B^a \bmod n$
- Bob computes $s = A^b \bmod n$

Exercise D1

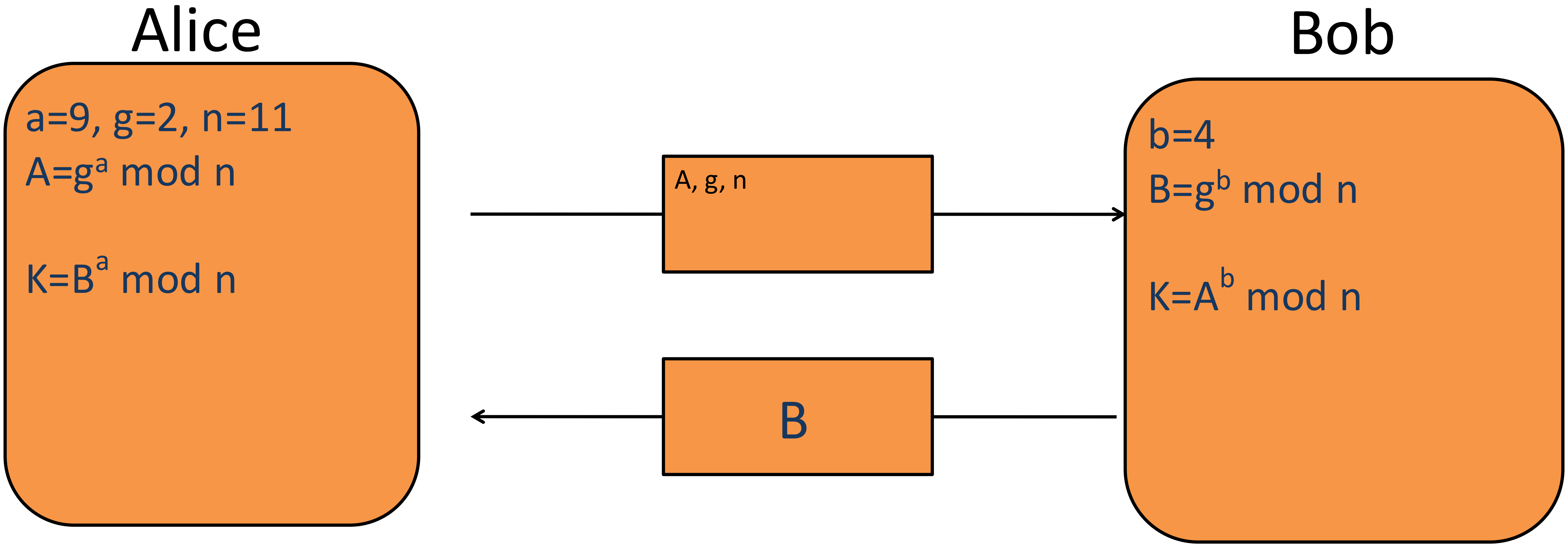
n=23 and base g=5.

Alice chooses a secret integer. a=6

Bob chooses a secret integer b=15



Exercise D2



RSA: Creating Public/Private Keypair

1. Choose two large prime numbers $p, q.$
   (e.g., 2048 bits each)

2. Compute $n = pq$, $\Phi = (p-1)(q-1)$

3. Choose e (with $1<e<\Phi$) that has no common factors with Φ. (e, Φ are "relatively prime").

4. Choose d such that $ed-1$ is exactly divisible by Φ.
   (in other words: $ed \bmod \Phi = 1$; or $d = e^{-1} \bmod \Phi$)

5. *Public* key is $(\underbrace{n,e}_{K_B^+})$. *Private* key is $(\underbrace{n,d}_{K_B^-})$.

Exercise E

- Using ***p=5, q=13***. Compute n, Φ, e, and d. Use the smallest value of e.

RSA: Creating Public/Private Keypair

1. Choose two large prime numbers ***p=5, q=13****.*
   (e.g., 1024 bits each)

2. Compute *n=65 and* Φ=48

3. Choose *e:*

4. Choose *d* such that *ed-1* is exactly divisible by Φ.
   (in other words: *ed* mod Φ = *1 ; or d = e   mod* Φ)

5. *Public* key is $(\underbrace{n,e}_{K_B^+})$=> . *Private* key is $(\underbrace{n,d}_{K_B^-})$=>.

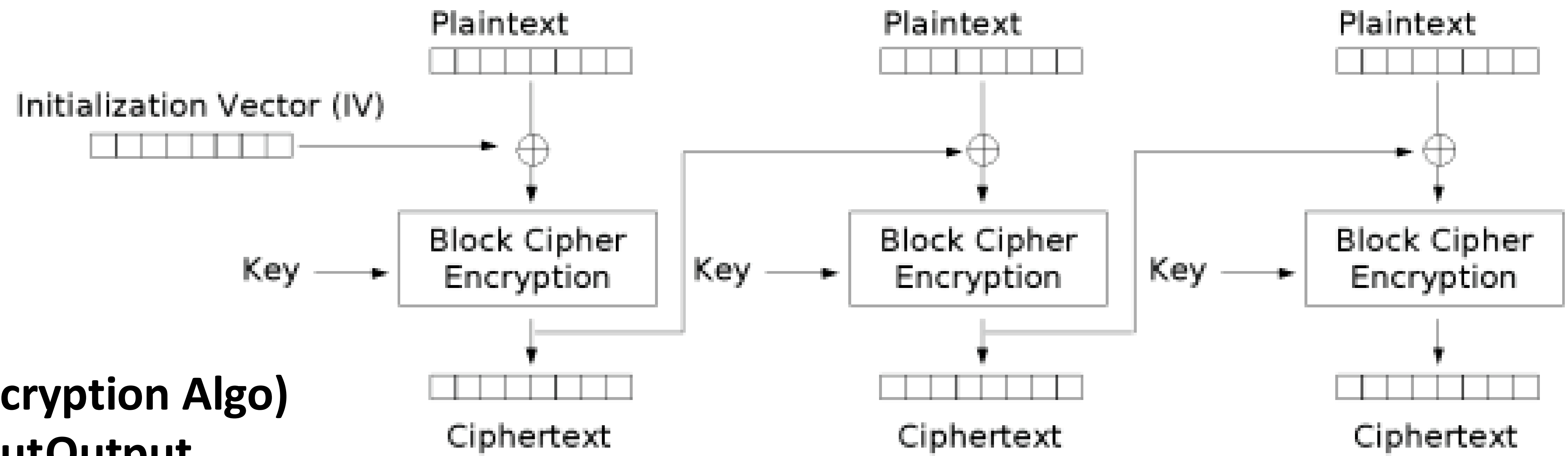
4. Choose *d* such that *ed-1* is exactly divisible by Φ.
(in other words: *ed* mod Φ = *1 ; or d = e   mod* Φ)

5. *Public* key is *(n,e)*. *Private* key is *(n,d)*.

RSA encryption & decryption

- Encrypt: m=10
- C = m^e mod n

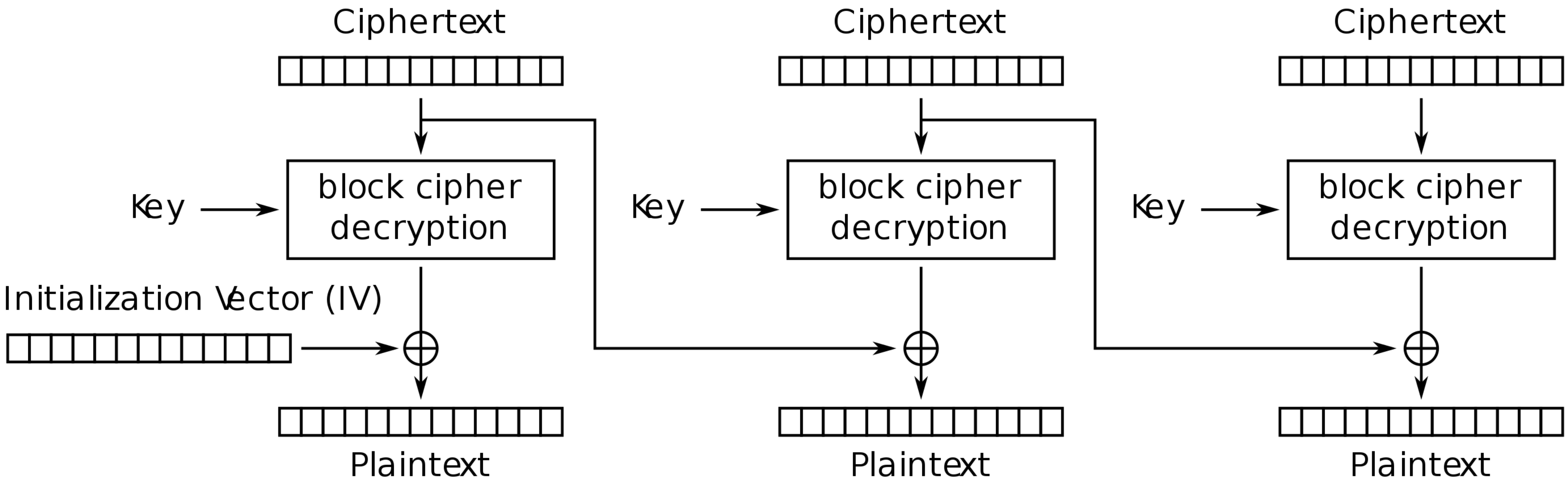
Exercise F (CBC)



cryption Algo)

ut	Output
0	110
1	111
0	100
1	101
0	011
1	010
0	001
1	000

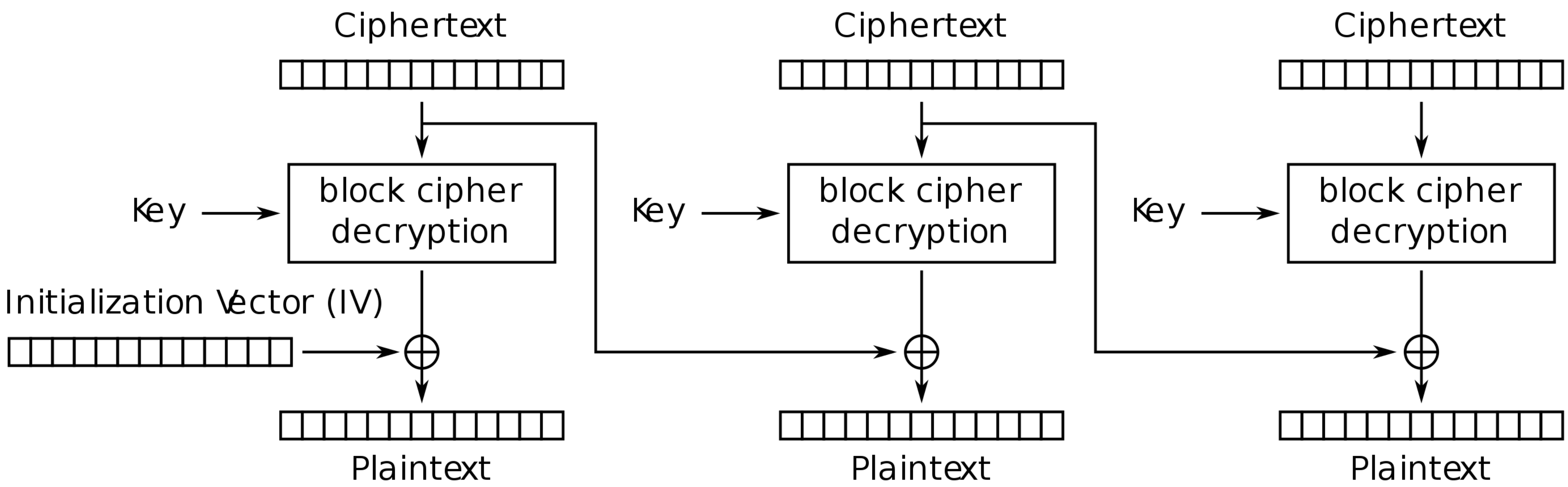
Encrypt: IV = 101   plaintext= 111 111 111



Cipher Block Chaining (CBC) mode decryption

Input	**Output**
000	110
001	111
010	100
011	101
100	011
101	010
110	001
111	000

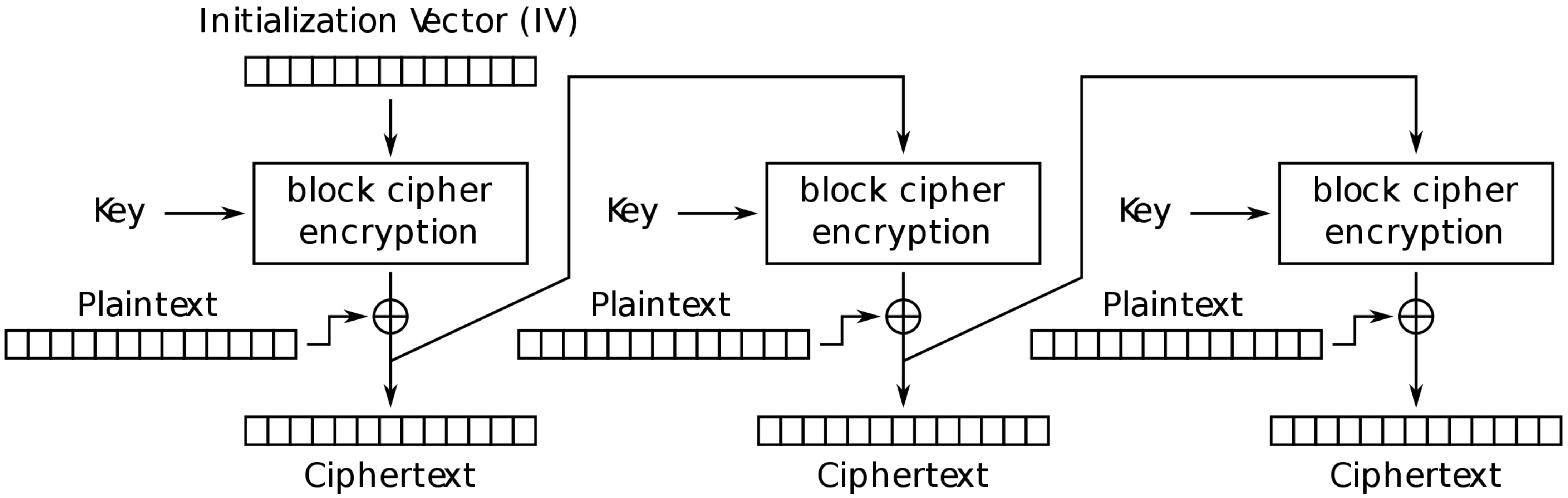
Ex. F2 CBC Decryption



Cipher Block Chaining (CBC) mode decryption

Input	**Output**
000	110
001	111
010	100
011	101
100	011
101	010
110	001
111	000

Exercise G (CFB)

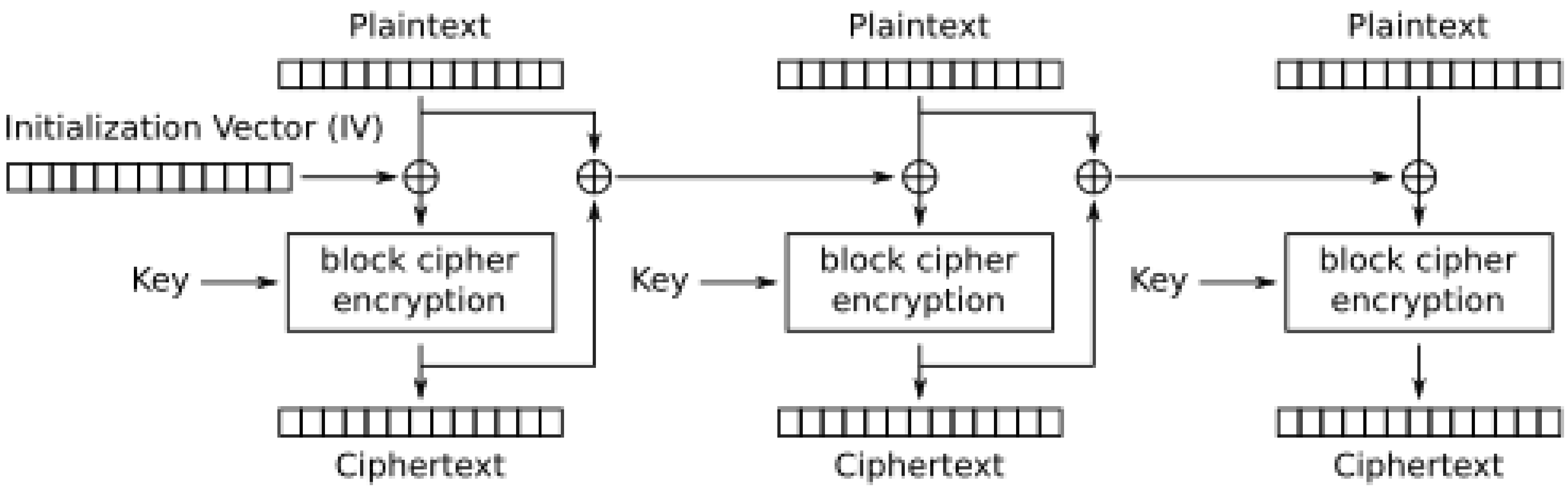


Cipher Feedback (CFB) mode encryption

Input	**Output**
000	110
001	111
010	100
011	101
100	011
101	010
110	001
111	000

IV=000 encrypt 001 001 001

Exercise G2 (PCBC)



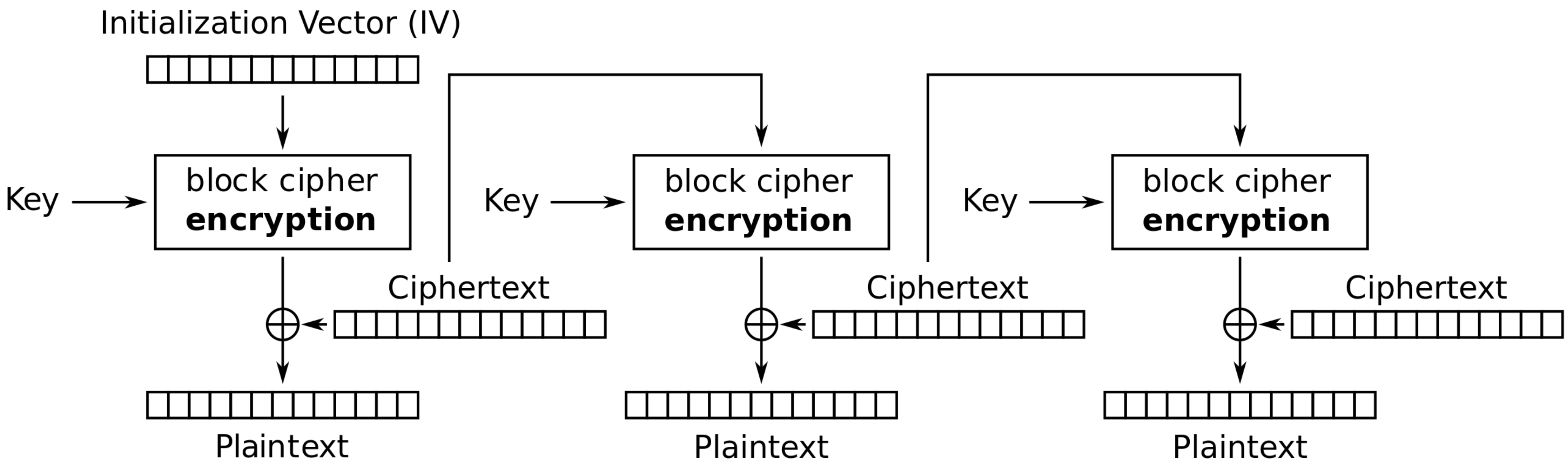
Propagating Cipher Block Chaining (PCBC) mode encryption

put	Output
0	110
1	111
0	100
1	101
0	011
1	010
0	001
1	000

IV=000 encrypt 001 001 001

Exercise H (Decrypt CFB)

Input	Output
000	110
001	111
010	100
011	101
100	011
101	010
110	001
111	000



Cipher Feedback (CFB) mode decryption